AMENDMENTS TO THE CLAIMS

This listing of claims will replace all prior versions, and listings, of claims in the application:

1. (Currently Amended) A method of specifying an asynchronous web service within a procedural programming environment, the method comprising:

providing a source code representation of at least a portion of web service logic of a web service to be offered by a server, the logic including at least one method declared to be a callback method;

identifying a member variable declared to implement said callback method to cause a compiler to generate a client proxy object for instantiation on the server for interacting asynchronously with the client using said callback method, and to assign the client proxy object to said member variable; and

specifying one or more declarative annotations associated with said callback method to cause a compiler to generate one or more persistent components for instantiation on the server to maintain conversational state related to the identified member variable.

2. (Original) The method of claim 1, wherein said callback method is declared inside a callback interface definition.

3. (Cancelled)

4. (Currently Amended) The method of claim 13, wherein the one or more declarative annotations indicate to the compiler whether the identified method is at least one of a start method, a continue method, and a finish method, wherein the start method applies to the start of a stateful conversation between the client and the web service, the continue method applies to the continuation of an ongoing stateful conversation between the client and the web service, and the

finish method applies to the completion of an ongoing stateful conversation between the client and the web service.

5. (Original) The method of claim 4, wherein when a method declared to be a start method is invoked at run-time, a new instance of a conversation is created, and a unique identifier is associated with that conversational instance to facilitate management of multiple simultaneous conversations.

6. (Original) The method of claim 4, wherein when a method declared to be a continue method or a finish method is invoked at run-time, a unique identifier is obtained and used to access a corresponding instance of a conversation.

7. (Original) The method of claim 6, wherein the unique identifier is provided by the client.

8. (Original) The method of claim 6, wherein when a method declared to be a finish method is invoked at run-time, the corresponding instance of the conversation is destroyed after processing by the web service logic.

9. (Currently Amended) The method of claim 1~~3~~, wherein the one or more declarative annotations indicate to the compiler whether the identified callback method is buffered, causing the compiler to instantiate one or more queues to temporarily store one or more asynchronous responses for delivery to the client when the client is able to receive the responses.

10.-11.(Cancelled)

12. (Previously Presented) In a server having an asynchronous web service, a method comprising:

receiving by the server a message from a client remotely disposed from the server requesting that the web service be invoked;

parsing by the server the message to identify the requested web service method in addition to a callback address indicating a location where the client is listening for callbacks from the web service;

storing by the server the callback address in association with a proxy object; and

invoking by the server the requested web service method, including passing programming language objects as parameters to the web service method, the programming language objects having been mapped from data representation language elements of the message.

13. (Previously Presented) The method of claim 12, further comprising:

identifying an instance identifier provided by the client indicating a particular instance of the client that is listening for callbacks from the web service; and

storing the instance identifier in association with the callback address and the proxy object.

14. (Original) The method of claim 13, wherein at least one of the callback address and the instance identifier is encapsulated in one or more request messages.

15. (Original) The method of claim 13, wherein the instance identifier is embedded within a conversation identifier used to access a corresponding instance of a conversation between the client and the web service.

16. (Original) The method of claim 14, wherein at least one of the callback address and the instance identifier is encapsulated in one or more SOAP message headers.

17. (Original) The method of claim 13, wherein the instance identifier is a GUID.

18. (Previously Presented) The method of claim 12, wherein invoking the requested web service method further comprises:

mapping programming language objects returned by the web service method onto representative data representation language elements;

packaging the representative data representation language elements according to one or more protocols used by the client in generating the requested message; and

transmitting the packaged data representation elements to the client in accordance with one or more protocols used by the client in transmitting the message to the web service.

19. (Previously Presented) The method of claim 13, further comprising:

generating an asynchronous response to the client in response to the message by invoking a method on the proxy object using a declared member variable, wherein invoking includes

passing programming language objects as parameters to said method,

mapping the programming language objects and method invocation onto representative data representation language elements;

packaging the representative data representation language elements, and callback instance identifier into an asynchronous response message, and

transmitting the asynchronous response message to the client at the callback address in accordance with one or more protocols used by the client in transmitting the original request message to the web service.

20. (Previously Presented) A method for specifying logic within a procedural programming environment for receiving a callback from an external web service, the method comprising:

identifying a member variable to be used for interacting with said external web service;

providing a method associated with said member variable, the method having a signature and containing logic for receiving said callback from said external web service; and

specifying one or more declarative annotations in association with said member variable to cause a compiler to generate one or more persistent components to maintain conversational state related to the external web service.

21. (Original) The method of claim 20, wherein said method is manually provided by a developer.

22. (Original) The method of claim 20, wherein the method signature is provided by an integrated development environment based on a specified service description file containing a declaration for said callback.

23. (Original) The method of claim 20, wherein said method is associated with said member variable using a method naming convention that utilizes the name of said member variable and the name of said callback.

24. (Cancelled)

25. (Previously Presented) The method of claim 20, wherein the one or more declarative annotations are specified within the source code.

26. (Previously Presented) The method of claim 20, wherein the one or more declarative annotations are specified outside of the source code.

27. (Previously Presented) The method of claim 20, wherein the one or more declarative annotations are manually specified by a developer.

28. (Previously Presented) The method of claim 20, wherein the one or more declarative annotations are automatically specified by an integrated development environment based upon input provided by a developer.

29. (Original) The method of claim 20, wherein asynchronous responses from the external web service are passed to said method associated with said member variable.

30. (Previously Presented) In a server offering a web service, a method comprising:

generating by the server a request to another external web service using a proxy object previously generated by a compiler based upon a service description file associated with the external web service, wherein the request includes a callback address to identify a location to which the external web service should return a response;

transmitting by the server the request as a request message to the external web service using one or more transmission protocols; and

receiving by the server an asynchronous response from the external web service.

31. (Original) The method of claim 30, wherein the callback address includes proxy object identifier.

32. (Original) The method of claim 30, wherein the callback address is included within one or more headers of the request message.

33. (Original) The method of claim 32, wherein the request message is a SOAP based message.

34. (Original) The method of claim 30, wherein the callback address comprises a URL identifying a location where the web service is listening for a response from the external web service.

35. (Original) The method of claim 30, wherein the request further includes a callback instance identifier representing a specific instance of the requesting web service to which asynchronous responses are to be routed.

36. (Previously Presented) The method of claim 35, wherein the callback instance identifier is included within one or more headers of the request message.

37. (Original) The method of claim 36, wherein the request message is a SOAP based message.

38-42. (Cancelled)

43. (Currently Amended) An article of manufacture comprising:

a storage medium having stored therein a plurality of programming instructions, which when executed provide a graphical interface to facilitate specification of an asynchronous web service within a procedural programming environment including

providing a source code representation of at least a portion of web service logic of a web service to be offered by a server, the logic including at least one method declared to be a callback method,

identifying a member variable declared to implement said callback method to cause a compiler to generate a client proxy object for instantiation on the server for interacting asynchronously with the

client using said callback method, and to assign the client proxy object to said member variable, and

specifying one or more declarative annotations associated with said callback method to cause a compiler to generate one or more persistent components for instantiation on the server to maintain conversational state related to the identified member variable.

44. (Original) The article of claim 43, wherein said callback method is declared inside a callback interface definition.

45. (Cancelled)

46. (Currently Amended) The article of claim 435, wherein the one or more declarative annotations indicate to the compiler whether the identified method is at least one of a start method, a continue method, and a finish method, wherein the start method applies to the start of a stateful conversation between the client and the web service, the continue method applies to the continuation of an ongoing stateful conversation between the client and the web service, and the finish method applies to the completion of an ongoing stateful conversation between the client and the web service.

47. (Original) The article of claim 46, wherein when a method declared to be a start method is invoked at run-time, a new instance of a conversation is created, and a unique identifier is associated with that conversational instance to facilitate management of multiple simultaneous conversations.

48. (Original) The article of claim 46, wherein when a method declared to be a continue method or a finish method is invoked at run-time, a unique identifier is obtained and used to access a corresponding instance of a conversation.

49. (Original) The article of claim 48, wherein the unique identifier is provided by the client.

50. (Original) The article of claim 48, wherein when a method declared to be a finish method is invoked at run-time, the corresponding instance of the conversation is destroyed after processing by the web service logic.

51. (Currently Amended) The article of claim 435, wherein the one or more declarative annotations indicate to the compiler whether the identified callback method is buffered, causing the compiler to instantiate one or more queues to temporarily store one or more asynchronous responses for delivery to the client when the client is able to receive the responses.

52.-53.(Cancelled)

54. (Previously Presented) An article of manufacture comprising:

a storage medium having stored therein a plurality of programming instructions designed to program a server to implement an asynchronous web service, which programming instructions when executed enable the server to

receive a message from a client requesting that a web service method be invoked;

parse the message to identify the requested web service method in addition to a callback address indicating a location where the client is listening for callbacks from the web service;

store the callback address in association with a proxy object; and

invoke the requested web service method, including passing programming language objects as parameters to the web service method, the programming language objects having been mapped from data representation language elements of the message.

55. (Previously Presented) The article of claim 54, wherein the programming instructions further enable the server to

identify an instance identifier provided by the client indicating a particular instance of the client that is listening for callbacks from the web service; and

store the instance identifier in association with the callback address and the proxy object.

56. (Original) The article of claim 55, wherein at least one of the callback address and the instance identifier is encapsulated in one or more request messages.

57. (Original) The article of claim 55, wherein the instance identifier is embedded within a conversation identifier used to access a corresponding instance of a conversation between the client and the web service.

58. (Original) The article of claim 56, wherein at least one of the callback address and the instance identifier is encapsulated in one or more SOAP message headers.

59. (Original) The article of claim 55, wherein the instance identifier is a GUID.

60. (Previously Presented) The article of claim 54, wherein the programming instructions to cause the server to invoke the requested web service further cause the server to

map programming language objects returned by the web service method onto representative data representation language elements;

package the representative data representation language elements according to one or more protocols used by the client in generating the requested message; and

transmit the packaged data representation elements to the client in accordance with one or more protocols used by the client in transmitting the message to the web service.

61. (Previously Presented) The article of claim 55, wherein the programming instructions further enable the server to generate an asynchronous response to the client in response to the message by invoking a method on the proxy object using a declared member variable, wherein invoking includes

passing programming language objects as parameters to said method,

mapping the programming language objects and method invocation onto representative data representation language elements;

packaging the representative data representation language elements, and callback instance identifier into an asynchronous response message, and

transmitting the asynchronous response message to the client at the callback address in accordance with one or more protocols used by the client in transmitting the original request message to the web service.

62. (Previously Presented) An article of manufacture comprising:

a storage medium having stored therein a plurality of programming instructions, which when executed provide a graphical interface to facilitate specification of an asynchronous web service within a procedural programming environment including

identifying a member variable to be used by a server for interacting with another external web service;

providing a method associated with said member variable, the method having a signature and containing logic for receiving said callback from said external web service; and

specifying one or more declarative annotations in association with said member variable to cause a compiler to generate one or

more persistent components to maintain conversational state related to the external web service.

63. (Original) The article of claim 62, wherein said method is manually provided by a developer.

64. (Original) The article of claim 62, wherein the method signature is provided by an integrated development environment based on a specified service description file containing a declaration for said callback.

65. (Original) The article of claim 62, wherein said method is associated with said member variable using a method naming convention that utilizes the name of said member variable and the name of said callback.

66. (Cancelled)

67. (Previously Presented) The article of claim 62, wherein the one or more declarative annotations are specified within the source code.

68. (Previously Presented) The article of claim 62, wherein the one or more declarative annotations are specified outside of the source code.

69. (Previously Presented) The article of claim 62, wherein the one or more declarative annotations are manually specified by a developer.

70. (Previously Presented) The article of claim 62, wherein the one or more declarative annotations are automatically specified by an integrated development environment based upon input provided by a developer.

71. (Original) The article of claim 62, wherein asynchronous responses from the external web service are passed to said method associated with said member variable.

72. (Previously Presented) An article of manufacture comprising:

a storage medium having stored therein a plurality of programming instructions designed to program a server to implement an asynchronous web service, which when executed enable the server to

generate a request to another external web service using a proxy object previously generated by a compiler based upon a service description file associated with the external web service, wherein the request includes a callback address to identify a location to which the external web service should return a response;

transmit the request as a request message to the external web service using one or more transmission protocols; and

receive an asynchronous response from the external web service.

73. (Original) The article of claim 72, wherein the callback address includes proxy object identifier.

74. (Original) The article of claim 72, wherein the callback address is included within one or more headers of the request message.

75. (Original) The article of claim 74, wherein the request message is a SOAP based message.

76. (Original) The article of claim 72, wherein the callback address comprises a URL identifying a location where the web service is listening for a response from the external web service.

77. (Original) The article of claim 72, wherein the request further includes a callback instance identifier representing a specific instance of the requesting web service to which asynchronous responses are to be routed.

78. (Original) The article of claim 77, wherein and the callback instance identifier is included within one or more headers of the request message.

79. (Original) The article of claim 78, wherein the request message is a SOAP based message.

80-84. (Cancelled)